Package ‘TExPosition’

February 19, 2015

Type Package
Title Two-table ExPosition
Version 2.6.10
Date 2013-12-09
Author Derek Beaton, Jenny Rieck, Cherise R. Chin Fatt, Herve Abdi
Maintainer Derek Beaton <exposition.software@gmail.com>
Description TExPosition is an extension of ExPosition for two table analyses, specifically, discriminant analyses.
License GPL-2
Depends prettyGraphs (>= 2.1.4), ExPosition (>= 2.0.0)
BugReports http://code.google.com/p/exposition-family/issues/list
NeedsCompilation no
Repository CRAN
Date/Publication 2013-12-11 16:07:11

R topics documented:

  TExPosition-package ........................................ 2
  calculateLVConstraints ................................... 3
  fastEucCalc ............................................. 4
  fii2fi ................................................ 4
  print.tepAssign ....................................... 5
  print.tepBADA ......................................... 6
  print.tepDICA .......................................... 6
  print.tepGPLS .......................................... 7
  print.tepGraphs ........................................ 7
  print.tepPLS .......................................... 8
  print.tepPLSCA ....................................... 8
  print.txtexoOutput .................................... 9
  R2 .................................................. 9
  tepBADA .............................................. 10
TExPosition-package

tepDICA ............................................................... 11
tepGPLS ............................................................... 13
tepGraphs .............................................................. 15
tepPLS ................................................................. 17
tepPLSCA ............................................................... 19
texpoDesignCheck .................................................... 20

Index 22

TExPosition-package  TExPosition: Two-table analyses with via ExPosition.

Description

TExPosition is two-table ExPosition and includes discriminant methods of the singular value decomposition (SVD). The core of TExPosition is ExPosition and the svd.

Details

Package: TExPosition
Type: Package
Version: 2.6.10
Date: 2013-12-00
Depends: R (>=2.15.0), prettyGraphs (>= 2.1.4), ExPosition (>= 2.0.0)
License: GPL-2
URL: http://www.utdallas.edu/~derekbeaton/software/ExPosition

Author(s)

Questions, comments, compliments, and complaints go to Derek Beaton <exposition.software@gmail.com>.

The following people are authors or contributors to TExPosition code, data, or examples: Derek Beaton, Jenny Rieck, Cherise Chin-Fatt, Francesca Filbey, and Hervé Abdi.

References

**calculateLVConstraints**


### See Also

tepBADA, tepPLS, tepGPLS, tepDICA, tepPLSCA

### Examples

#For more examples, see each individual function (as noted above).

```r

```

calculateLVConstraints

```

description

Calculates constraints for plotting latent variables.

**Usage**

```r
calculateLVConstraints(results, x_axis=1, y_axis=2, constraints=NULL)
```

**Arguments**

- `results` results (with $lx and $ly) from TExPosition (i.e., $TExPosition.Data)
- `x_axis` which component should be on the x axis?
- `y_axis` which component should be on the y axis?
- `constraints` if available, axis constraints for the plots (determines end points of the plots).

**Value**

Returns a list with the following items:

- `$constraints` axis constraints for the plots (determines end points of the plots).

**Author(s)**

Derek Beaton
fastEucCalc  

Description

Fast Euclidean distance calculations.

Usage

fastEucCalc(x, c)

Arguments

x  
a set of points.

c  
a set of centers.

Details

This function is especially useful for discriminant analyses. The distance from each point in x to each point in c is computed and returned as a nrow(x) x nrow(c) matrix.

Value

a distance matrix

Euclidean distances of each point to each center are returned.

Author(s)

Hervé Abdi, Derek Beaton

fii2fi  

Description

All computations between individual factor scores (fii) and group factor scores (fi).

Usage

fii2fi(DiGN, fii, fi)

Arguments

DiGN  
a dummy-coded design matrix

fii  
a set of factor scores for individuals (rows)

fi  
a set of factor scores for rows
Value

A list of values containing:

- **distances**: Euclidean distances of all rows to each category center
- **assignments**: an assignment matrix (similar to DESIGN) where each individual is assigned to the closest category center
- **confusion**: a confusion matrix of how many items are assigned (and mis-assigned) to each category

Author(s)

Hervé Abdi, Derek Beaton

---

print.tepAssign

*Print assignment results*

**Description**

Print assignment results.

**Usage**

```r
## S3 method for class 'tepAssign'
print(x, ...)
```

**Arguments**

- **x**: an list that contains items to make into the tepAssign class.
- **...**: inherited/passed arguments for S3 print method(s).

**Author(s)**

Derek Beaton, Cherise Chin-Fatt
print.tepBADA

Description
Print tepBADA results.

Usage
```r
## S3 method for class 'tepBADA'
print(x,...)
```

Arguments
- `x` an list that contains items to make into the tepBADA class.
- `...` inherited/passed arguments for S3 print method(s).

Author(s)
Derek Beaton, Cherise Chin-Fatt

print.tepDICA

Description
Print tepDICA results.

Usage
```r
## S3 method for class 'tepDICA'
print(x,...)
```

Arguments
- `x` an list that contains items to make into the tepDICA class.
- `...` inherited/passed arguments for S3 print method(s).

Author(s)
Derek Beaton, Cherise Chin-Fatt
print.tepGPLS

```r
print.tepGPLS

Description
Print tepGPLS results.

Usage
## S3 method for class 'tepGPLS'
print(x, ...)

Arguments
x an list that contains items to make into the tepGPLS class.
...
inherited/passed arguments for S3 print method(s).

Author(s)
Derek Beaton, Cherise Chin-Fatt
```

print.tepGraphs

```r
print.tepGraphs

Description
Print tepGraphs results.

Usage
## S3 method for class 'tepGraphs'
print(x, ...)

Arguments
x an list that contains items to make into the tepGraphs class.
...
inherited/passed arguments for S3 print method(s).

Author(s)
Derek Beaton, Cherise Chin-Fatt
```
print.tepPLS  

Print tepPLS results

Description

Print tepPLS results.

Usage

```r
## S3 method for class 'tepPLS'
print(x,...)
```

Arguments

- `x`: an list that contains items to make into the tepPLS class.
- `...`: inherited/passed arguments for S3 print method(s).

Author(s)

Derek Beaton, Cherise Chin-Fatt

print.tepPLSCA  

Print tepPLSCA results

Description

Print tepPLSCA results.

Usage

```r
## S3 method for class 'tepPLSCA'
print(x,...)
```

Arguments

- `x`: an list that contains items to make into the tepPLSCA class.
- `...`: inherited/passed arguments for S3 print method(s).

Author(s)

Derek Beaton, Cherise Chin-Fatt
**print.texpoOutput**

*Print TExPosition results*

---

**Description**

Print TExPosition results.

**Usage**

```r
## S3 method for class 'texpoOutput'
print(x, ...)
```

**Arguments**

- `x` an list that contains items to make into the texpoOutput class.
- `...` inherited/passed arguments for S3 print method(s).

**Author(s)**

Derek Beaton, Cherise Chin-Fatt

---

**R2**

* R-squared computations

---

**Description**

A function to compute R-squared for BADA and DICA

**Usage**

```r
R2(group.masses, di, ind.masses = NULL, dii)
```

**Arguments**

- `group.masses` a masses matrix for the groups
- `di` a set of squared distances of the groups
- `ind.masses` a masses matrix for the individuals
- `dii` a set of squared distances for the individuals

**Value**

- `R2` An R-squared

**Author(s)**

Jenny Rieck, Derek Beaton
tepBADA

Barycentric Discriminant Analysis

Description

Barycentric Discriminant Analysis (BADA) via TExPosition.

Usage

tepBADA(DATA, scale = TRUE, center = TRUE, DESIGN = NULL, make_design_nominal = TRUE, group.masses = NULL, weights = NULL, graphs = TRUE, k = 0)

Arguments

DATA
original data to perform a BADA on.

scale
a boolean, vector, or string. See expo.scale for details.

center
a boolean, vector, or string. See expo.scale for details.

DESIGN
a design matrix to indicate if rows belong to groups. Required for BADA.

make_design_nominal
a boolean. If TRUE (default), DESIGN is a vector that indicates groups (and will be dummy-coded). If FALSE, DESIGN is a dummy-coded matrix.

group.masses
a diagonal matrix or column-vector of masses for the groups.

weights
a diagonal matrix or column-vector of weights for the column items.

graphs
a boolean. If TRUE (default), graphs and plots are provided (via tepGraphs)

k
number of components to return.

Details

Note: BADA is a special case of PLS (tepPLS, tepGPLS) wherein DATA1 are data and DATA2 are a group-coded disjunctive matrix. This is also called mean-centered PLS (Krishnan et al., 2011).

Value

See epGPCA (and also corePCA) for details on what is returned. In addition to the values returned:

fii
factor scores computed for supplemental observations

dii
squared distances for supplemental observations

rii
cosines for supplemental observations

assign
a list of assignment data. See fii2fi and R2

lx
latent variables from DATA1 computed for observations

ly
latent variables from DATA2 computed for observations
Discriminant Correspondence Analysis

Author(s)
Derek Beaton

References

See Also
corePCA, epPCA, epGPCA, epMDS
For Matlab code: http://utd.edu/~derekbeaton/attachments/Software/matlab/MuSuBADA_V3.zip

Examples
```r
data(bada.wine)
bada.res <- tepBADA(bada.wine$data,scale=FALSE,DESIGN=bada.wine$design,make_design_nominal=FALSE)
```

tepDICA

Description
Discriminant Correspondence Analysis (DICA) via TExPosition.

Usage
```r
tepDICA(DATA, make_data_nominal = FALSE, DESIGN = NULL, make_design_nominal = TRUE, group.masses = NULL, weights = NULL, symmetric = TRUE, graphs = TRUE, k = 0)
```
Arguments

DATA
original data to perform a DICA on. Data can be contingency (like CA) or
categorical (like MCA).

make_data_nominal
a boolean. If TRUE (default), DATA is recoded as a dummy-coded matrix. If
FALSE, DATA is a dummy-coded matrix.

DESIGN
a design matrix to indicate if rows belong to groups. Required for DICA.

make_design_nominal
a boolean. If TRUE (default), DESIGN is a vector that indicates groups (and
will be dummy-coded). If FALSE, DESIGN is a dummy-coded matrix.

group.masses
a diagonal matrix or column-vector of masses for the groups.

weights
a diagonal matrix or column-vector of weights for the column it

symmetric
a boolean. If TRUE (default) symmetric factor scores for rows.

graphs
a boolean. If TRUE (default), graphs and plots are provided (via tepGraphs)
k
number of components to return.

Details

If you use Hellinger distance, it is best to set symmetric to FALSE.

Note: DICA is a special case of PLS-CA (tepPLSCA) wherein DATA1 are data and DATA2 are
a group-coded disjunctive matrix.

Value

See epCA (and also coreCA) for details on what is returned. In addition to the values returned:

fii
factor scores computed for supplemental observations

dii
squared distances for supplemental observations

rii
cosines for supplemental observations

assign
a list of assignment data. See fii2fi and R2

lx
latent variables from DATA1 computed for observations

ly
latent variables from DATA2 computed for observations

Author(s)

Derek Beaton, Hervé Abdi

References


See Also
coreCA, epCA, epMCA
For MatLab code: http://utd.edu/~herve/HerveAbdi_MatlabPrograms4MUDICA.zip For additional R code (with inference tests): http://utdallas.edu/~dfb090020/attachments/MuDiCA.zip

Examples
data(dica.wine)
dica.res <- tepDICA(dica.wine$data, DESIGN=dica.wine$design, make_design_nominal=FALSE)

tepGPLS Generalized Partial Least Squares

Description

Generalized Partial Least Squares (GPLS) via TExPosition. GPLS is to PLS (tepPLS) as PCA epPCA is to GPCA epGPCA. The major difference between PLS and GPLS is that GPLS allows the use of weights for the columns of each data set (just like GPCA).

Usage
tepGPLS(DATA1, DATA2,
center1 = TRUE, scale1 = "SS1",
center2 = TRUE, scale2 = "SS1",
DESIGN = NULL, make_design_nominal = TRUE,
weights1 = NULL, weights2 = NULL,
graphs = TRUE, k = 0)
Arguments

- **DATA1**  
  Data matrix 1 (X)

- **DATA2**  
  Data matrix 2 (Y)

- **center1**  
  a boolean, vector, or string to center DATA1. See `expo.scale` for details.

- **scale1**  
  a boolean, vector, or string to scale DATA1. See `expo.scale` for details.

- **center2**  
  a boolean, vector, or string to center DATA2. See `expo.scale` for details.

- **scale2**  
  a boolean, vector, or string to scale DATA2. See `expo.scale` for details.

- **DESIGN**  
  a design matrix to indicate if rows belong to groups.

- **make_design_nominal**  
  a boolean. If TRUE (default), DESIGN is a vector that indicates groups (and will be dummy-coded). If FALSE, DESIGN is a dummy-coded matrix.

- **weights1**  
  a weight vector (or diag matrix) for the columns of DATA1.

- **weights2**  
  a weight vector (or diag matrix) for the columns of DATA2.

- **graphs**  
  a boolean. If TRUE (default), graphs and plots are provided (via `tepGraphs`)

- **k**  
  number of components to return.

Details

This implementation of Partial Least Squares is a symmetric analysis. It was first described by Tucker (1958), again by Bookstein (1994), and has gained notoriety in Neuroimaging from McIntosh et al., (1996). This particular implementation allows the user to provide weights for the columns of both **DATA1** and **DATA2**.

Value

See **epGPCA** (and also **corePCA**) for details on what is returned. In addition to the values returned:

- **1x**  
  latent variables from DATA1 computed for observations

- **1y**  
  latent variables from DATA2 computed for observations

- **data1.norm**  
  center and scale information for DATA1

- **data1.norm**  
  center and scale information for DATA2

Author(s)

Derek Beaton

References


**See Also**
corePCA, epPCA, epGPCA, tepPLS, tepPLSCA, tepBADA, tepDICA

**Examples**
```r
data(beer.tasting.notes)
data1<-beer.tasting.notes$data[,1:8]
data2<-beer.tasting.notes$data[,9:16]
gpls.res <- tepGpls(data1,data2)
```

**Description**
TEXPosition plotting function which is an interface to prettyGraphs.

**Usage**
```r
tepGraphs(res, x_axis = 1, y_axis = 2,
  tepPlotInfo = NULL, DESIGN = NULL,
  fi.col = NULL, fi.pch = NULL, fii.col = NULL, fii.pch = NULL,
  fj.col = NULL, fj.pch = NULL, col.offset = NULL,
  constraints = NULL, lv.constraints = NULL,
  xlab = NULL, ylab = NULL, main = NULL,
  lvPlots = TRUE, lvAgainst = TRUE,
  contributionPlots = TRUE, correlationPlotter = TRUE,
  showHulls = 1, biplots = FALSE, graphs = TRUE)
```

**Arguments**
- **res**: results from TExPosition
- **x_axis**: which component should be on the x axis?
- **y_axis**: which component should be on the y axis?
- **tepPlotInfo**: A list ($Plotting.Data$) from tepGraphs or TExPosition.
- **DESIGN**: A design matrix to apply colors (by pallete selection) to row items
- **fi.col**: A matrix of colors for the group items. If NULL, colors will be selected.
- **fi.pch**: A matrix of pch values for the group items. If NULL, pch values are all 21.
tepGraphs

fii.col: A matrix of colors for the row items (observations). If NULL, colors will be selected.
fii.pch: A matrix of pch values for the row items (observations). If NULL, pch values are all 21.
fj.col: A matrix of colors for the column items. If NULL, colors will be selected.
fj.pch: A matrix of pch values for the column items. If NULL, pch values are all 21.
col.offset: A numeric offset value. Is passed to createColorVectorsByDesign.
constraints: Plot constraints as returned from prettyPlot. If NULL, constraints are selected.

lv.constraints: Plot constraints for latent variables. If NULL, constraints are selected.
xlab: x axis label
ylab: y axis label
main: main label for the graph window
lvPlots: a boolean. If TRUE, latent variables (X, Y) are plotted. If FALSE, latent variables are not plotted.
lvAgainst: a boolean. If TRUE, latent variables (X, Y) are plotted against each other. If FALSE, latent variables are plotted like factor scores.

contributionPlots: a boolean. If TRUE (default), contribution bar plots will be created.
correlationPlotter: a boolean. If TRUE (default), a correlation circle plot will be created. Applies to PCA family of methods (CA is excluded for now).

showHulls: a value between 0 and 1 to make a peeled hull at that percentage. All values outside of 0-1 will not plot any hulls.
biplots: a boolean. If FALSE (default), separate plots are made for row items ($fii and $fi) and column items ($fj). If TRUE, row ($fii and $fi) and column ($fj) items will be on the same plot.

graphs: a boolean. If TRUE, graphs are created. If FALSE, only data associated to plotting (e.g., constraints, colors) are returned.

Details

tepGraphs is an interface between TExPosition and prettyGraphs.

Value

The following items are bundled inside of $Plotting.Data:

$fii.col: the colors that are associated to the individuals (row items; $fii).
$fii.pch: the pch values associated to the individuals (row items; $fii).
$fi.col: the colors that are associated to the groups ($fi).
$fi.pch: the pch values associated to the groups ($fi).
$fj.col: the colors that are associated to the column items ($fj).
$fj.pch: the pch values associated to the column items ($fj).
$constraints: axis constraints for the plots (determines end points of the plots).
Partial Least Squares (PLS) via TExPosition.

**Usage**

```r
tepPLS(DATA1, DATA2, 
center1 = TRUE, scale1 = "SS1", center2 = TRUE, scale2 = "SS1", 
DESIGN = NULL, make_design_nominal = TRUE, 
graphs = TRUE, k = 0)
```

**Arguments**

- **DATA1**: Data matrix 1 (X)
- **DATA2**: Data matrix 2 (Y)
- **center1**: a boolean, vector, or string to center DATA1. See `expo.scale` for details.
- **scale1**: a boolean, vector, or string to scale DATA1. See `expo.scale` for details.
- **center2**: a boolean, vector, or string to center DATA2. See `expo.scale` for details.
- **scale2**: a boolean, vector, or string to scale DATA2. See `expo.scale` for details.
- **DESIGN**: a design matrix to indicate if rows belong to groups.
- **make_design_nominal**: a boolean. If TRUE (default), DESIGN is a vector that indicates groups (and will be dummy-coded). If FALSE, DESIGN is a dummy-coded matrix.
- **graphs**: a boolean. If TRUE (default), graphs and plots are provided (via `tepGraphs`)
- **k**: number of components to return.

**Examples**

```r
# this is for TExPosition's iris data
data(ep.iris)
bada.iris <- tepBADA(ep.iris$data, DESIGN = ep.iris$design, make_design_nominal = FALSE)
# there are only 2 components, not 3.
bada.iris.plotting.data.biplot <- tepGraphs(bada.iris, x_axis=1, y_axis=2, biplots=TRUE)
```
Details

This implementation of Partial Least Squares is a symmetric analysis. It was first described by Tucker (1958), again by Bookstein (1994), and has gained notoriety in Neuroimaging from McIntosh et al., (1996).

Value

See epGPCA (and also corePCA) for details on what is returned. In addition to the values returned:

- 1x: latent variables from DATA1 computed for observations
- 1y: latent variables from DATA2 computed for observations
- data1.norm: center and scale information for DATA1
- data1.norm: center and scale information for DATA2

Author(s)

Derek Beaton

References


See Also

corePCA, epPCA, epGPCA, tepBADA, tepGPLS, tepPLSCA

Examples

data(beer.tasting.notes)
data1<-beer.tasting.notes$data[,1:8]
data2<-beer.tasting.notes$data[,9:16]
pls.res <- tepPLS(data1,data2)
Partial Least Squares-Correspondence Analysis (PLSCA) via TExPosition.

**Usage**

```r
tepPLSCA(DATA1, DATA2, make_data1_nominal = FALSE, make_data2_nominal = FALSE, DESIGN = NULL, make_design_nominal = TRUE, weights1=NULL, weights2 = NULL, symmetric = TRUE, graphs = TRUE, k = 0)
```

**Arguments**

- **DATA1**: Data matrix 1 (X), must be categorical (like MCA) or in disjunctive code see `make_data1_nominal`.
- **DATA2**: Data matrix 2 (Y), must be categorical (like MCA) or in disjunctive code see `make_data2_nominal`.
- **make_data1_nominal**: a boolean. If TRUE (default), DATA1 is recoded as a dummy-coded matrix. If FALSE, DATA1 is a dummy-coded matrix.
- **make_data2_nominal**: a boolean. If TRUE (default), DATA2 is recoded as a dummy-coded matrix. If FALSE, DATA2 is a dummy-coded matrix.
- **DESIGN**: a design matrix to indicate if rows belong to groups.
- **make_design_nominal**: a boolean. If TRUE (default), DESIGN is a vector that indicates groups (and will be dummy-coded). If FALSE, DESIGN is a dummy-coded matrix.
- **weights1**: a diagonal matrix or column-vector of weights for the columns of DATA1
- **weights2**: a diagonal matrix or column-vector of weights for the columns of DATA2
- **symmetric**: a boolean. If TRUE (default) symmetric factor scores for rows.
- **graphs**: a boolean. If TRUE (default), graphs and plots are provided (via `tepGraphs`)
- **k**: number of components to return.

**Details**

This implementation of Partial Least Squares is for two categorical data sets (Beaton et al., 2013), and based on the PLS method proposed by Tucker (1958) and again by Bookstein (1994).
Value

See epCA (and also coreCA) for details on what is returned. In addition to the values returned:

\[
\begin{align*}
\mathbf{w}_1 & \quad \text{Weights for columns of DATA1, replaces } m \text{ from coreCA}. \\
\mathbf{w}_2 & \quad \text{Weights for columns of DATA2, replaces } w \text{ from coreCA}. \\
\mathbf{l}_x & \quad \text{latent variables from DATA1 computed for observations} \\
\mathbf{l}_y & \quad \text{latent variables from DATA2 computed for observations}
\end{align*}
\]

Author(s)

Derek Beaton, Hervé Abdi

References


See Also

coreCA, epCA, epMCA, tepDICA

Examples

data(snp$sdruguse)
plsc.a.res <- tepPLSCA(snp$sdruguse$DATA1,snp$sdruguse$DATA2,
make_data1_nominal=TRUE,make_data2_nominal=TRUE)

texpoDesignCheck  texpoDesignCheck

Description

TExPosition’s DESIGN matrix check function. Calls into ExPosition’s designCheck.
Usage

texpoDesignCheck(DATA = NULL, DESIGN = NULL, make_design_nominal = TRUE, force_bary=FALSE)

Arguments

DATA original data that should be matched to a design matrix
DESIGN a column vector with levels for observations or a dummy-coded matrix
make_design_nominal a boolean. Will make DESIGN nominal if TRUE (default).
force_bary a boolean. If TRUE, it forces the check for barycentric methods (tepDICA, tepBADA). If FALSE, designCheck is performed.

Details

For BADA & DICA, execution stops if:
1. DESIGN has more columns (groups) than observations, 2. DESIGN has only 1 column (group), or 3. DESIGN has at least 1 occurrence where an observation is the only observation in a group (i.e., colSums(DESIGN)==1 at least once).

Value

DESIGN dummy-coded design matrix

Author(s)

Derek Beaton
Index

*Topic graphs
  tepGraphs, 15

*Topic misc
  calculateLVConstraints, 3
  fastEuCalc, 4
  fii2fi, 4
  R2, 9
  tepGraphs, 15
  texpoDesignCheck, 20

*Topic multivariate
  fii2fi, 4
  R2, 9
  tepBADA, 10
  tepDICA, 11
  tepGLS, 13
  tepGraphs, 15
  tepPLS, 17
  tepPLSCA, 19
  TExPosition-package, 2

*Topic package
  TExPosition-package, 2

*Topic print
  print.tepAssign, 5
  print.tepBADA, 6
  print.tepDICA, 6
  print.tepGLS, 7
  print.tepGraphs, 7
  print.tepPLS, 8
  print.tepPLSCA, 8
  print.texpoOutput, 9

  calculateLVConstraints, 3
  coreCA, 12, 13, 20
  corePCA, 10, 11, 14, 15, 18
  createColorVectorsByDesign, 16

  designCheck, 20, 21

  epCA, 12, 13, 20
  epGPCA, 10, 11, 13–15, 18

  epMCA, 13, 20
  epMDS, 11
  epPCA, 11, 13, 15, 18
  expo.scale, 10, 14, 17
  ExPosition, 2

  fastEuCalc, 4
  fii2fi, 4, 10, 12

  prettyGraphs, 15–17
  prettyPlot, 16
  print.tepAssign, 5
  print.tepBADA, 6
  print.tepDICA, 6
  print.tepGLS, 7
  print.tepGraphs, 7
  print.tepPLS, 8
  print.tepPLSCA, 8
  print.texpoOutput, 9

  R2, 9, 10, 12

  svd, 2

  tepBADA, 3, 10, 15, 18
  tepDICA, 3, 11, 15, 20
  tepGLS, 3, 10, 13, 18
  tepGraphs, 10, 12, 14, 15, 17, 19
  tepPLS, 3, 10, 13, 15, 17
  tepPLSCA, 3, 12, 15, 18, 19
  texpoDesignCheck, 20
  TExPosition, 16
  TExPosition (TExPosition-package, 2
  TExPosition-package, 2

22